

Relacionamento entre Objetos

Luiz Henrique de Campos Merschmann
Departamento de Computação Aplicada
Universidade Federal de Lavras

luiz.hcm@ufla.br

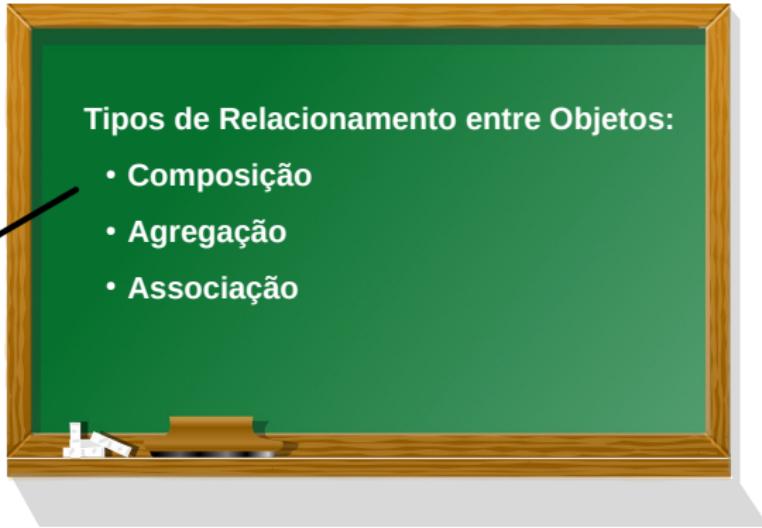


Na Aula de Hoje



Tipos de Relacionamento entre Objetos:

- Composição
- Agregação
- Associação



Relacionamento entre Objetos

Na programação orientada a objetos, os **objetos se relacionam**, ou seja, **enviam mensagens** uns para os outros.

- ▶ Esse envio de mensagens se dá quando um objeto faz chamadas a métodos públicos de outros objetos.



Relacionamento entre Objetos



Em OO os sistemas são desenvolvidos utilizando-se as classes. A partir delas os objetos são instanciados e podem começar a trocar mensagens entre si.

Sistemas OO dependem de como os objetos se relacionam. Portanto, o planejamento/modelagem das classes se baseia nesses relacionamentos.



Pensando nos Relacionamentos...

Quais objetos podem existir em um software de uma pizzaria?

Como você acha que esses objetos se relacionam?



O que cada um tem a ver com o outro?

Quem envia mensagem para quem?

Podemos pensar que existem, por exemplo, os objetos:

- ▶ Pizza, Bebida, Vendedor, Cliente etc.
- ▶ Um pedido pode ser formado por um conjunto de itens (pizza e bebida) e estar vinculado a um cliente e um vendedor.

Pensando nos Relacionamentos...

Quais objetos podem existir em um software como o SIG-UFLA?



Pense novamente como os objetos se relacionam.

O que cada um tem a ver com o outro?

Podemos pensar que existem, por exemplo, os objetos:

- ▶ Aluno, Turma, Professor, Disciplina etc.
- ▶ Uma turma pode ser formada por um conjunto de alunos e estar vinculada a uma disciplina.

Pensando nos Relacionamentos...

Por fim, quais objetos podem existir em um software de um banco?



E nesse caso, como os objetos se relacionam?

O que cada um tem a ver com o outro?

Podemos pensar que existem, por exemplo, os objetos:

- ▶ Agência, Cliente, Funcionário, Contas etc.
- ▶ Um banco pode ter diversas agências.
- ▶ Um cliente pode ter várias contas (corrente, poupança...) e estar vinculado a uma agência.

Relacionamento entre Objetos

Com os exemplos anteriores, podemos perceber que os softwares orientados a objetos dependem em grande parte de como os objetos se relacionam.

Portanto, temos que entender agora os **tipos de relacionamento** que podem existir entre objetos:

- ▶ Composição.
- ▶ Agregação.
- ▶ Associação.



Relacionamento: Banco x Agências

Qual é o relacionamento entre um banco e suas agências?



Podemos dizer que um banco
tem várias agências.

Relacionamento: Pedido x Itens

Qual é o relacionamento entre um pedido e os itens de um pedido?



Novamente, podemos dizer que um pedido **tem vários** itens.

Relacionamento: Livro x Capítulos

Qual é o relacionamento entre um livro e os capítulos?



Mais uma vez podemos dizer que
um livro **tem vários** capítulos.

Composição

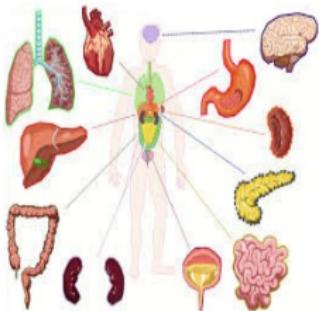


O que esses três exemplos têm em comum?

- ▶ O fato de que um objeto tem (um ou mais) objetos de outra classe.

Esse tipo de relacionamento é denominado **composição**.

Composição



A composição se caracteriza por ser um relacionamento: **Todo/Parte**

Ou seja, há uma classe representando o **todo** e classes “satélites” funcionando como **partes**.

Além disso, há outra característica importante:

- ▶ Nos exemplos vistos:
 - ▶ Se eu não tenho um banco, não preciso das agências.
 - ▶ Se não há um pedido, não faz sentido pensar em itens de pedido.
 - ▶ Se eu destruir um livro, não faz sentido os capítulos existirem.
- ▶ Resumindo: o objeto contido **não faz sentido** sem o objeto que o contém.

Ou seja, o todo **controla o tempo de vida** das partes

Composição – Exemplo de Código



Suponha que precisamos implementar em Java um sistema que tenha as classes Carro, Motor e Porta.

- ▶ Podemos dizer que um carro tem um motor.
- ▶ Um carro tem 4 portas.

Como isso pode ser implementado?

Composição – Exemplo de Código

```
class Carro{  
    private Motor motor;  
    private Porta[] portas;  
  
    public Carro(){  
        motor = new Motor();  
        portas = new Porta[4];  
        for(int i = 0; i < 4; i++){  
            portas[i] = new Porta();  
        }  
    }  
  
    public Motor getMotor(){  
        return motor.clone();  
    }  
  
    public Porta[] getPorta(){  
        return portas.clone();  
    }  
}
```

Como o motor e as portas são partes do carro, eles devem ser atributos do mesmo.

O carro controla o tempo de vida do motor e das portas, portanto é ele que cria esses objetos.

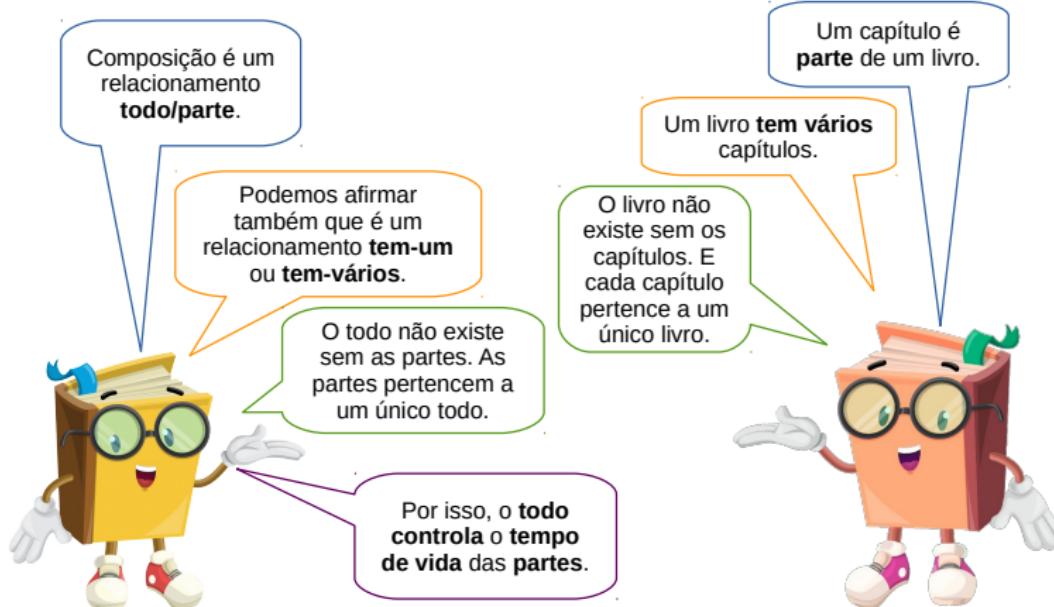
É importante que a classe não tenha um método get que retorne a referência do objeto (isso quebra o encapsulamento).

Como ninguém tem referência para os objetos porta e motor, esses objetos só podem ser destruídos junto com o carro. Portanto, o tempo de vida está completamente controlado.

Fonte: Material do Prof. Júlio

Resumindo...

Composição



Relacionamento: Grupo de Estudo x Alunos

Qual é o relacionamento entre um grupo de estudo e os alunos?



Um grupo de estudo **tem vários** alunos. Desse modo, um aluno é parte de um grupo de estudo.

Então isso é uma composição, certo?

Quando o grupo de estudo acabar, os alunos deixam de existir?



Agregação



Essa também é uma relação todo/parte, porém, nesse caso dizemos que a parte não é exclusiva do todo.

- ▶ Ou seja, não é o grupo de estudo que controla o tempo de vida dos alunos.
- ▶ Dizendo de outro modo, **não** é o **todo** que controla o **tempo de vida** das **partes**.

Repare:

- ▶ Os alunos continuam existindo ainda que o grupo de estudo termine! Ufa...ainda bem.

Esse tipo de relacionamento é denominado **agregação**.

Agregação – Exemplo de Código

```
class GrupoEstudo{  
    private ArrayList<Aluno> alunos;  
  
    public GrupoEstudo(){  
        alunos = new ArrayList<Aluno>();  
    }  
  
    public void cadastrarAluno(Aluno aluno){  
        alunos.add(aluno);  
    }  
}
```

Como o grupo de estudo tem alunos, os alunos devem ser atributos do grupo de estudo.

Mas se o grupo de estudo deixar de existir, os alunos continuam existindo! Portanto, essa é uma relação de agregação.

Na agregação o **todo** não controla o tempo de vida das **partes**. Portanto, uma maneira de tratar isso é o grupo de estudo não criar alunos (e sim recebê-los de outro lugar).

Resumindo...

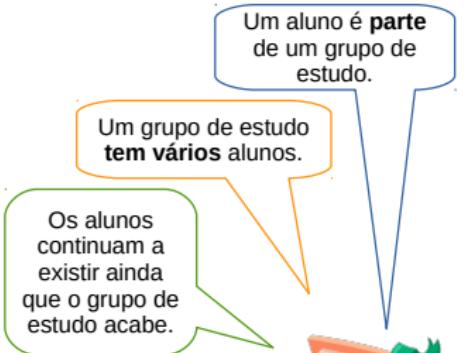
Agregação



Podemos enxergar a agregação com um tipo menos rigoroso de composição.

Os dois relacionamentos são **todo/parte** ou **tem-um**.

Mas na agregação, o todo não controla o tempo de vida das partes.



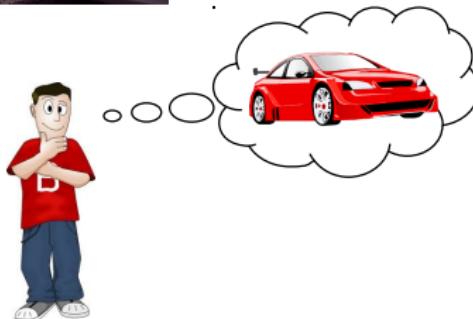
Relacionamento: Carro x Pessoa

Suponha que uma locadora de veículos resolva criar um sistema para controlar suas atividades.



Para esse sistema, foram criadas as classes **Carro** e **Pessoa**.

Qual é o relacionamento entre essas classes?



Associação



A pessoa utiliza o carro e o carro transporta a pessoa.

- ▶ Não faz sentido dizer que pessoa é parte do carro ou que carro é formado por pessoas, concorda?
- ▶ Ou seja, **não é** um relacionamento **todo/parte**.

Em outras palavras, temos duas classes distintas onde uma faz uso da outra.

Esse tipo de relacionamento é denominado **associação**.

Associação – Exemplo de Código

```
class Pessoa{  
    private CarteiraMotorista cnh;  
  
    public Pessoa(){  
    }  
  
    public void obterCarteira(CarteiraMotorista cnh){  
        this.cnh = cnh;  
    }  
  
    public void viajar(Carro carro, int distancia){  
        ...  
        carro.acelerar();  
        ...  
        carro.parar();  
    }  
}
```

Carteira de motorista não é parte de uma pessoa e nem o contrário. Portanto, é uma associação.

Mas existe um relacionamento de propriedade, ou seja, um motorista possui uma CNH.

Do mesmo modo, carro não é parte de pessoa e nem o contrário. Então, é associação também.

Mas existe um relacionamento, ou seja, uma pessoa viaja em um carro.

Repare que temos duas formas de implementação aqui. Uma com atributo e outra não.

O que importa de fato é que, na associação, um objeto chama métodos do outro.

Fonte: Material do Prof. Júlio

Resumindo...

Composição, Agregação e Associação



Mas se além de ser **todo/parte**, as partes **não existem sem o todo**, temos uma relação de **composição**.

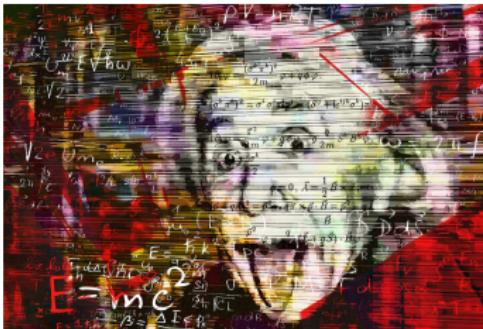
Se dois objetos se relacionam, mas um **não é parte** do outro, então eles possuem uma relação de **associação**.

Se a relação passa a ter características **todo/parte**, mas o todo **não controla o tempo de vida** das partes, temos uma **agregação**.



A relatividade...

ATENÇÃO: A classificação é **relativa**!



A **classificação** do relacionamento entre objetos é **relativa**. Ou seja, para definirmos se é uma composição, agregação ou associação temos que **analisar o contexto**.



Exemplo da relatividade...

Utilizamos anteriormente como exemplo de **composição** a relação entre **carro, motor e portas**.



Mas o que podemos dizer ao ler a notícia abaixo?

FÓRMULA 1

Fernando Alonso usará motor do carro reserva na corrida



Nesse caso percebemos que o motor existe independentemente do carro, ou seja, não é o carro que controla o tempo de vida do motor.

O espanhol Fernando Alonso irá largar na última posição do grid de largada para o Grande Prêmio da Malásia, segunda etapa do Mundial de Fórmula 1 que será disputado na madrugada

Portanto, agora temos uma **agregação**.

Perguntas?

